SECTION 8
ACTIVITY SHEET

FOUNDATION TRAINING GUIDE

Suitable for Blue Prism version 6.2 or greater

Blue Prism Limited, 2 Cinnamon Park, Birchwood, WA2 0XP, United Kingdom
Registered in England: Reg. No. 4260035. Tel: +44 870 879 3000. Web: www.blueprism.com

Section 8 Activity Sheet

In Section 8, you will learn about the key concepts associated with Exception Handling and about best practice for building Exception Handling logic into Processes and Business Objects.

Errors may occur at any point within a Blue Prism Solution, so ensuring comprehensive Exception Handling measures are in place, means that Processes can keep running in the event of such errors.

To achieve this, it's important to know how to identify when errors may be likely to occur - and how they should be handled when they do. Exception Handling can be built into Process Diagrams and Business Objects via the use of new Stages, that you will be introduced to in this Section.

SECTION 8 ACTIVITY 1a — Video 8.1

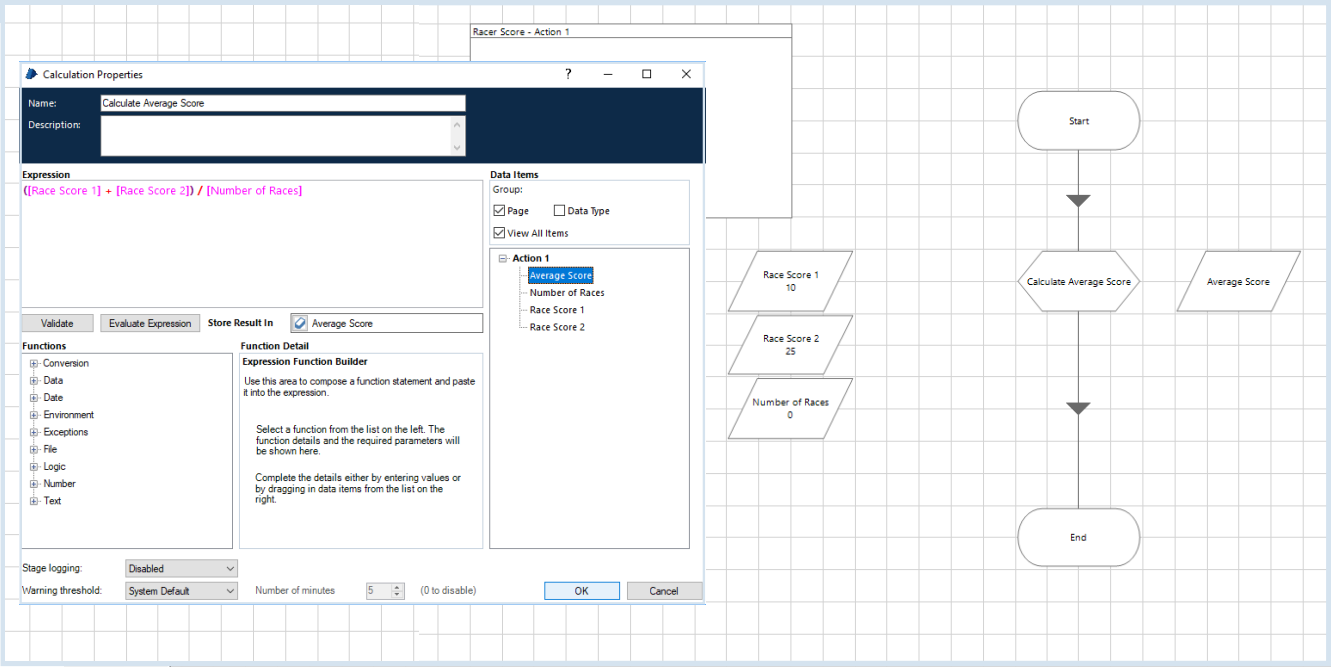
In this activity, you will create a new Business Object that calculates the average score of a competitor across a series of races. By building some simple calculation logic that uses values within Data Items.

You will then induce an error that triggers an Exception, by entering an unviable value into one of the Data Items. Finally, you will add a Recover and a Resume Stage to your Object Diagram, to catch the Exception.

Create a new Business Object that calculates the average score of a competitor across a series of races and induce a deliberate error.

- For this activity, you will be working in Object Studio.
- Create a new Business Object and name it *Racer Score*.
- To the *Action 1* Page, add three Data Items with *Number* Data Type.
  - Two for the race scores – *Race 1 Score* and *Race 2 Score*, each with an Initial value of your choosing.
  - And one for the *Number of Races*.
- The *Number of Races* Data Item will contain a deliberate error – an Initial Value of *0*. This will be recognized as an Exception, as Calculation Stages cannot divide *Number* values by *0*.
- Add a Calculation Stage that adds the two *Race Scores* together and divides the result by the *Number of Races*. *([Race Score 1]) + [Race Score2]) / [Number of Races]*.
- Type *Average Score* into the Store Result In field and click the auto-generate icon to the left of the Store Result In field, to generate a new Data Item of that name *(Average Score)* within your Action Diagram.

- Link all the Stages together.
- Save, reset and run the Action Page.
- An error message window should appear, to inform you that an Exception has occurred. Click *OK* and reset the Action Page.

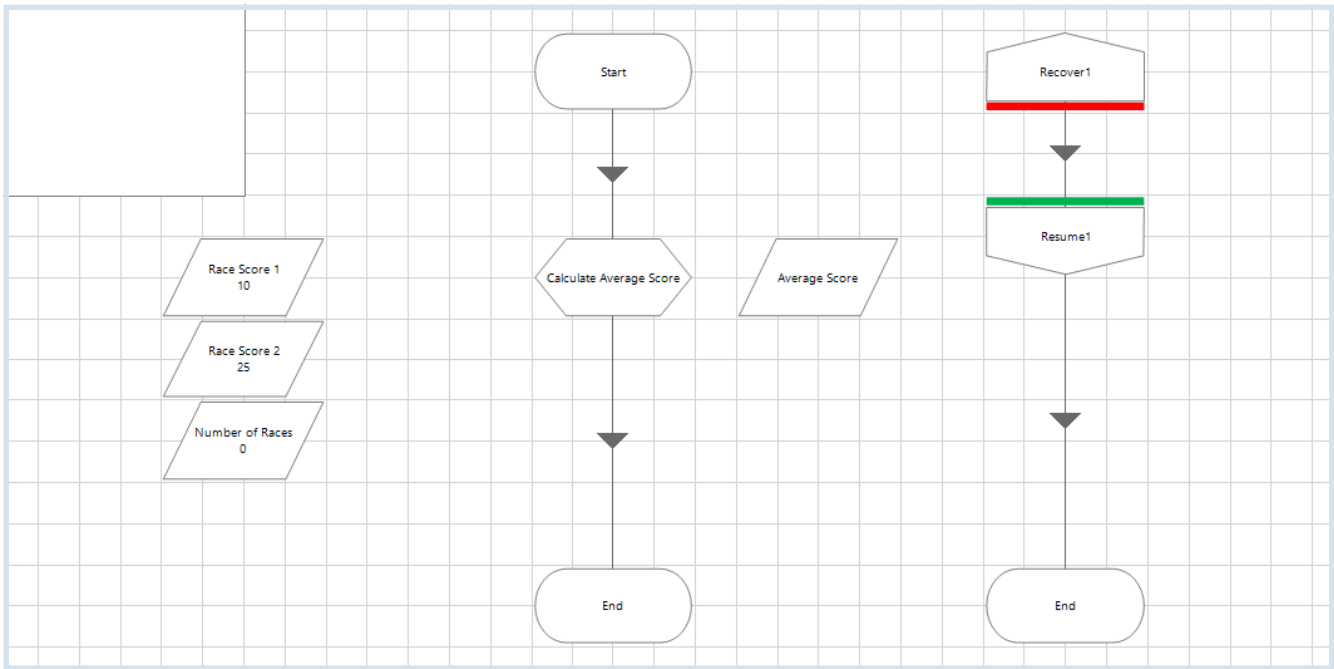


SECTION 8 ACTIVITY 1b

Video 8.1

Build some Recovery Logic to catch the Exception.

- To the Action Page, add a Recover Stage.
- Add a Resume Stage.
- Add a second End Stage.
- Link the Stages together. Though please note; Recover Stages do not have an inbound link as they only come into play when an Exception occurs.
- Save, reset and run the Action Page again, to observe the results. This time, the Recover Stage should catch the Exception and the Process should flow between the Recover Stage and the Resume Stage.



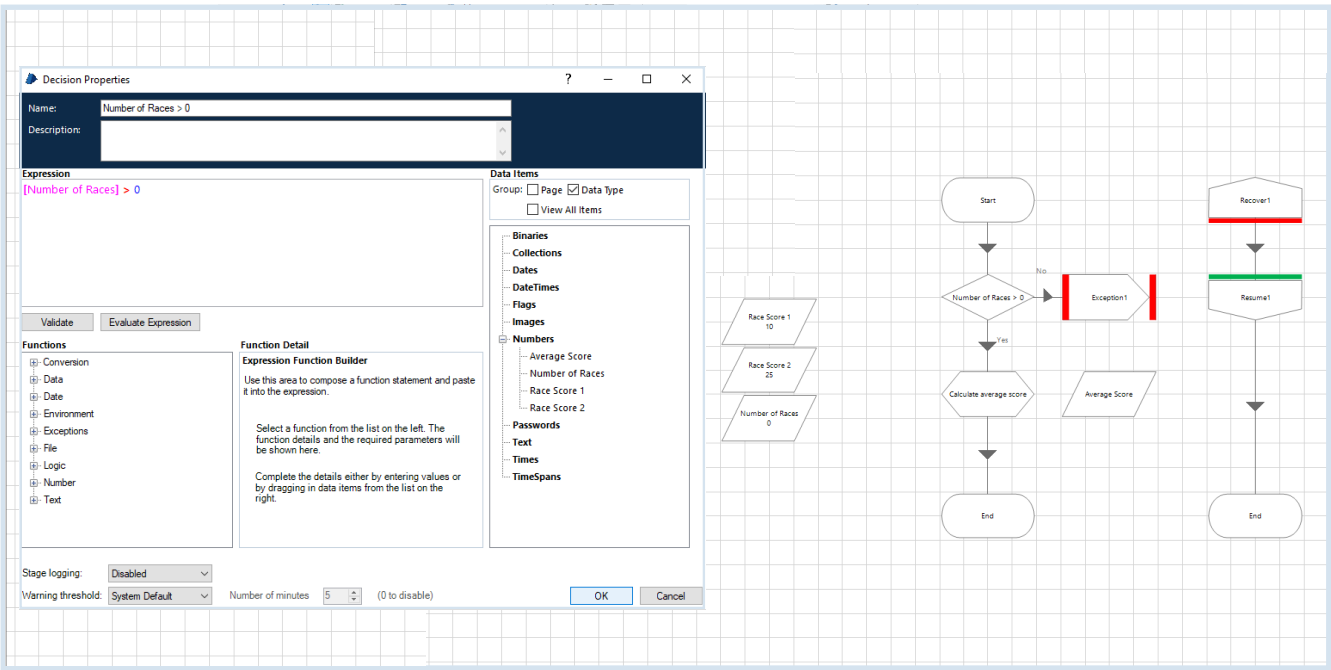
SECTION 8 ACTIVITY 2a

Video 8.2

In this activity, you will learn how to use an Exception Stage within an Action Diagram to Throw an Exception. You will then learn how to configure Exception Type and Exception Detail information within the Exception Properties.

Set up an Exception Stage to Throw an Exception into Recovery Mode.

- For this activity, you will be working in the *Racer Score* Business Object.
- On the *Action 1* Page, add a Decision Stage to the diagram and create an Expression *[Number of Races] > 0* to check whether the value in the *Number of Races* Data Item is greater than *0*.
- Add an Exception Stage to provide a path for the *No* branch of the Decision Stage and link all the Stages together.
- Save, reset and run the Action Page, to watch the Exception Stage Throw the Exception into Recovery Mode.

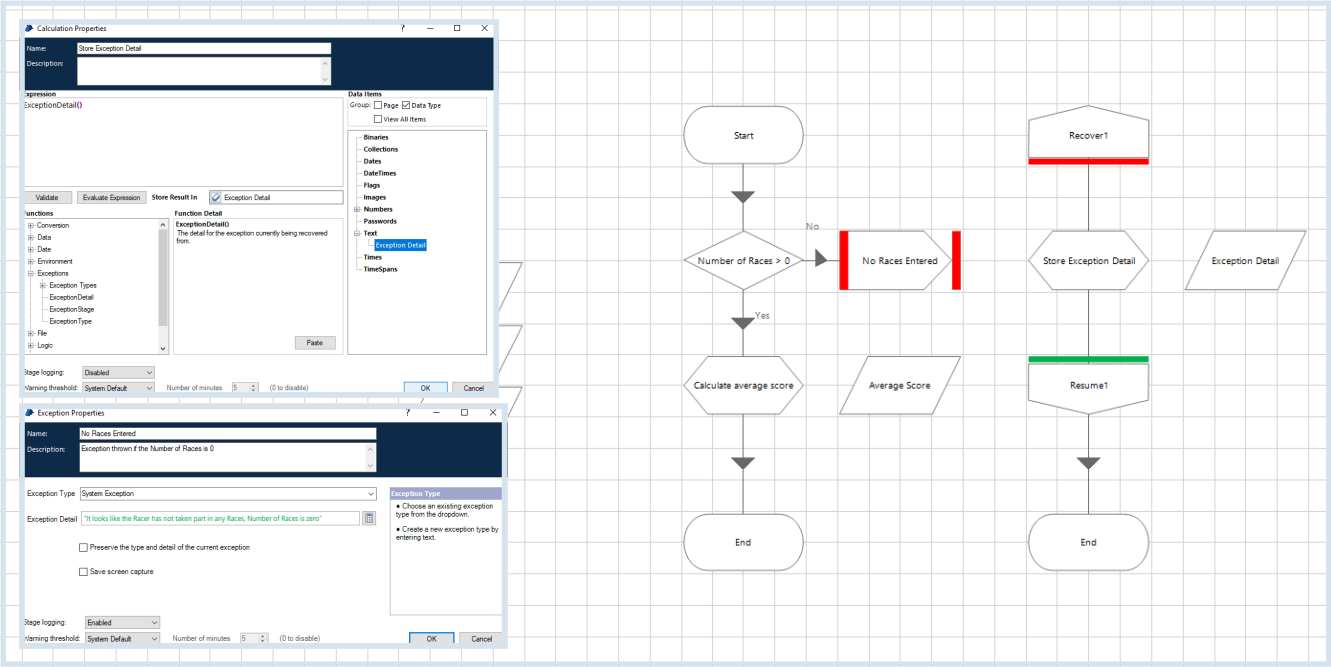


SECTION 8 ACTIVITY 2b

Video 8.2

Set up the Exception Stage to Throw an Exception and store the Exception Detail within a Data Item.

- Open the Exception Stage properties.
- Give the Exception a name *No Races Entered* and a description *Exception thrown if the Number of Races is 0*.
- In this example, *System Exception* is the logical Exception Type definition.
- The Exception Detail value is an Expression, so must be a Data Item or wrapped in quotation marks. Provide the feedback *"It looks like the Racer has not taken part in any Races, Number of Races is zero"*.
- Add a Calculation Stage between the Recover Stage and the Resume Stage and link them all together.
- Open the Calculation Stage properties and *drag* the *ExceptionDetail()* function into the Expression Area.
- Type *Exception Detail* into the Store Result In field. Click the autogenerate icon to the left of the Store Result In field, to create a new Data Item of that name *(Exception Detail)* within the Action Page.
- Within a Stage properties window, Exception Functions such as *ExceptionDetail()* and *ExceptionType()* can only be used between the Recover and Resume Stages.
- Reset and save the Action Page.



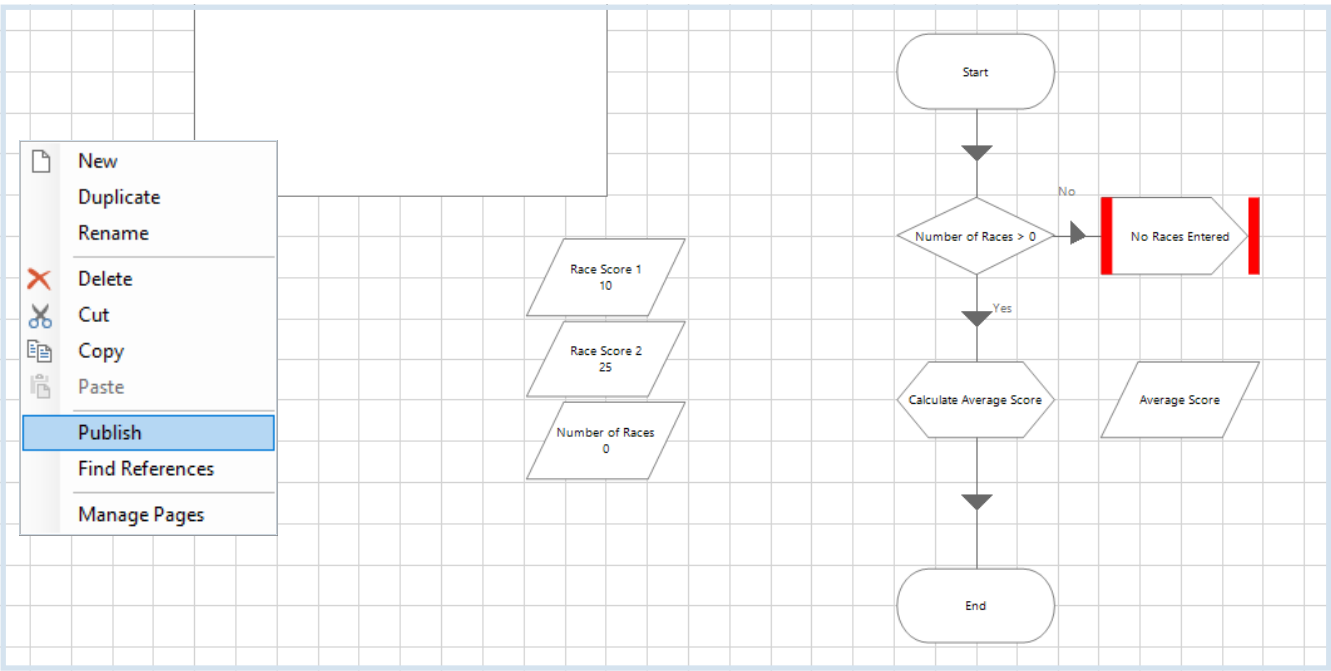
SECTION 8 ACTIVITY 3a

Video 8.3

In this activity, you will create a Process that uses the ***Racer Score*** Business Object. You will then experiment with the different ways that Exceptions can Bubble up through Business Objects and Process Pages.

Remove the Recovery logic and create a Process that uses the *Racer Score* Business Object.

- For this activity, you'll be working in both Process Studio and Object Studio.
- Open the *Racer Score* Business Object.
- *Cut* the *Recovery logic* from the *Action 1* Page.
- Publish the *Action 1* Page, then save and close the Business Object.

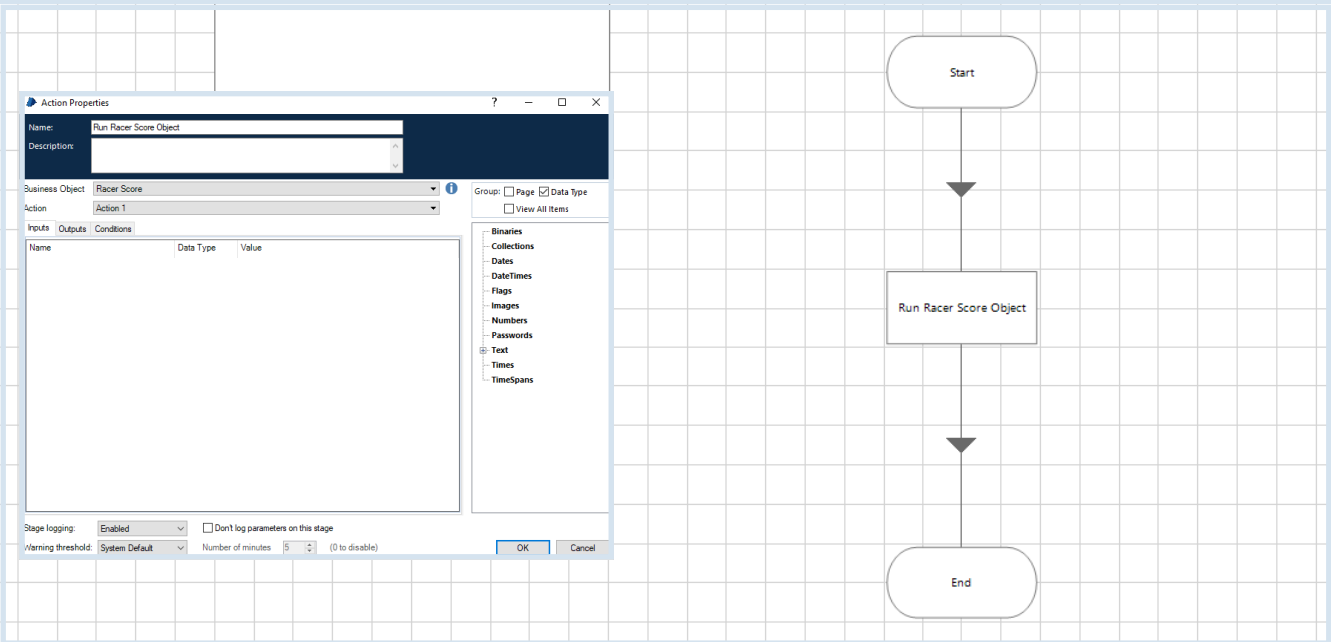


SECTION 8 ACTIVITY 3b

Video 8.3

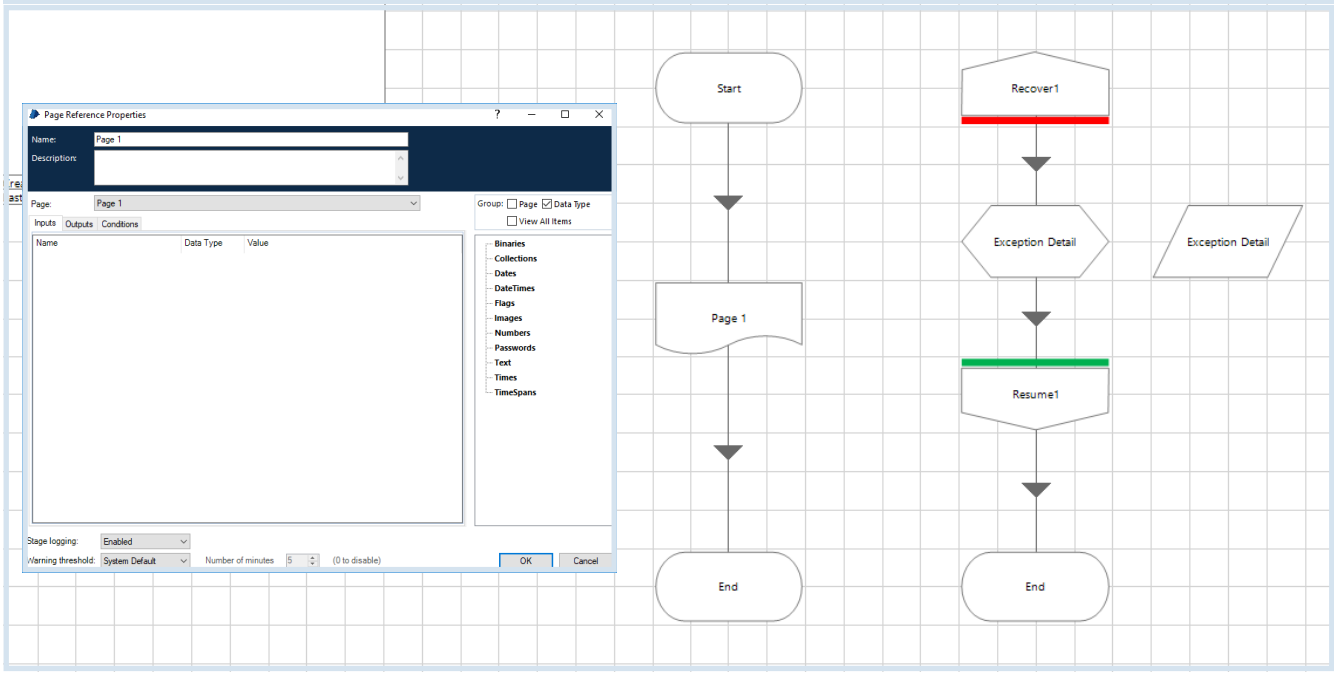
Experiment by causing the Exception to Bubble up from the Business Object, to the calling Process.

- Create a new Process called *Racer Scores*.
- To the Main Page, add an Action Stage that uses the *Racer Score* Business Object. Link the Stages together.
- Reset, save and run the Process. Close the error message window, then reset the Process.



Experiment by allowing the Exception to Bubble up from the Business Object, through the calling Page and then up to the Main Page within the Process.

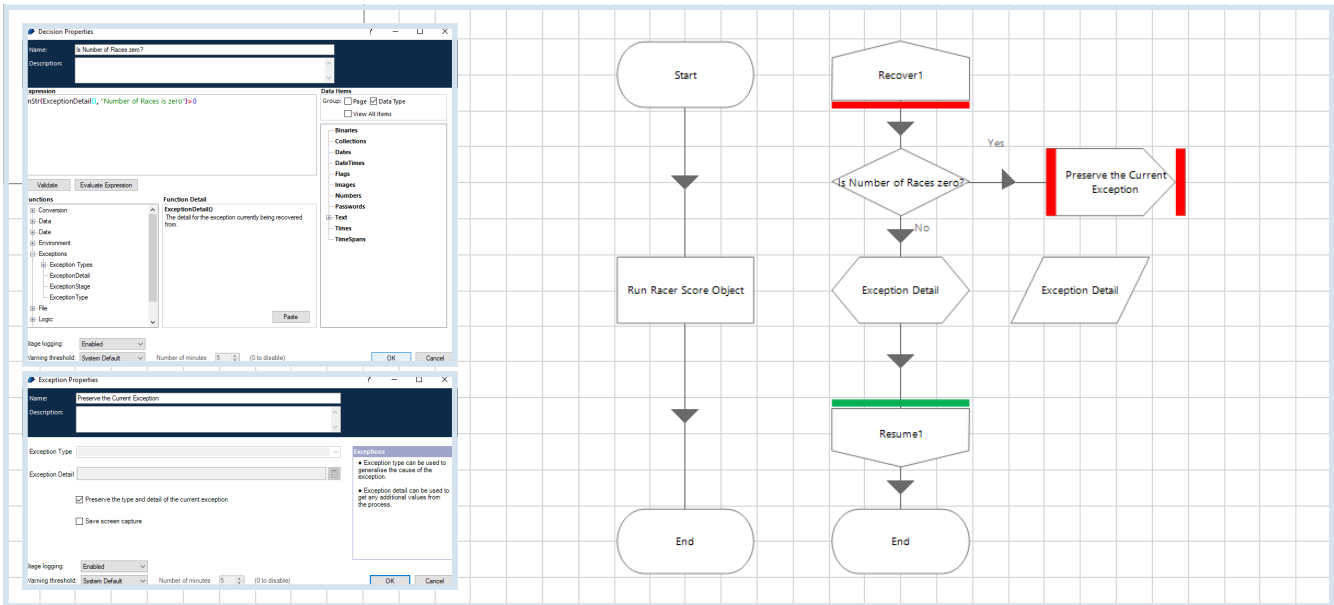
- Add another Page to the *Racer Scores* Process called *Page 1*.
- On the Main Page, *paste* the Recovery logic you cut from the *Racer Score* Business Object.
- *Cut* the Action Stage *(Run Racer Score Object)* from the Main Page and add a Page Reference Stage in its place. Link it to the Start and End Stages.
- *Paste* the *Run Racer Score Object* Action Stage onto *Page 1*. and link it to the Start and End Stages.
- Reset, save and run the Process.



In this activity, you will add a Decision Stage and a new Exception Stage inside Recovery Mode, that will Re-Throw any Exceptions that it can't handle up to the Main Page of the Process. You will also learn how to use the Preserve Type and Detail checkbox to retain the Exception information generated by the Business Object, as it Bubbles up towards the Main Page of the Process.

Add a Decision Stage and a new Exception Stage inside Recovery Mode, to Re-Throw any Exceptions it can't handle up to the Main Page of the Process.

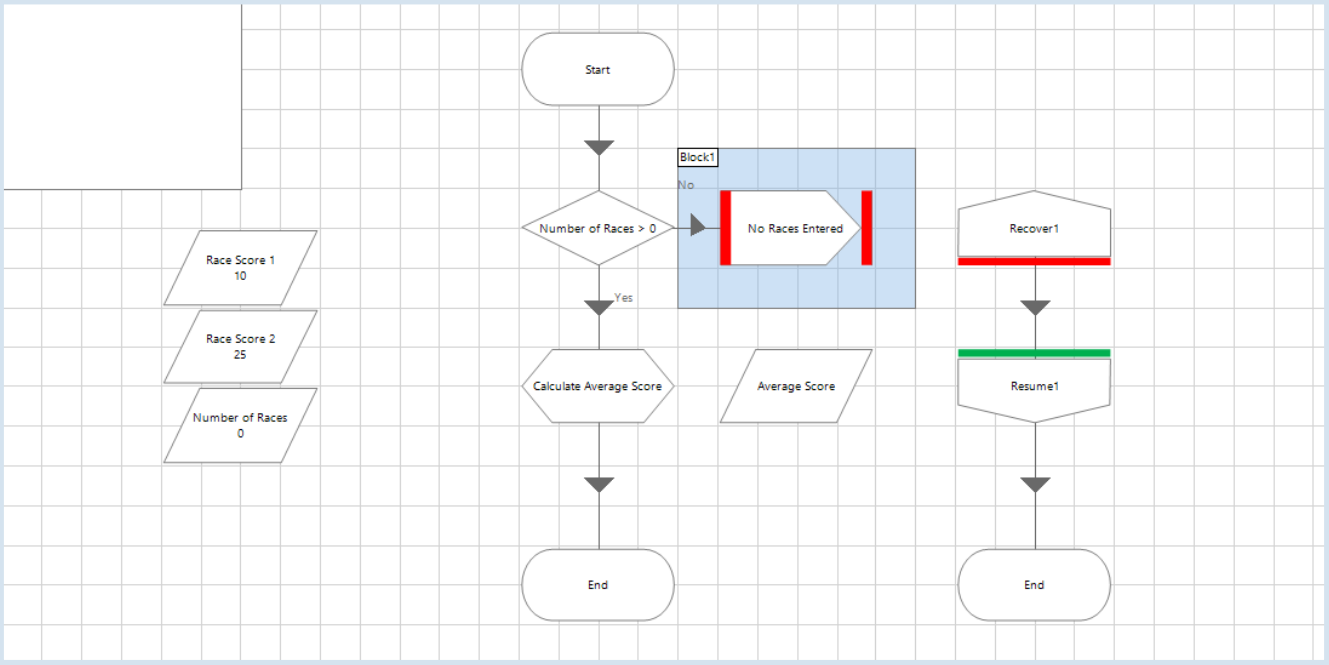
- For this activity, you'll be working in the *Racer Scores Process*, in Process Studio.
- *Cut* the Recovery logic from the Main Page and *paste* it onto *Page 1*.
- On *Page 1*, add a Decision Stage between the Recover and Calculation Stages.
- Create an Expression in the Decision properties to check whether the recovered Exception is due to a zero in the *Number of Races* Data Item. There are numerous ways to do this, here's one possible Expression: *InStr(ExceptionDetail(), "Number of Races is zero")>0*.
- Add an Exception Stage to the Process Diagram and link the *Yes* branch from the Decision Stage to the new Exception Stage. Link the *No* branch from the Decision Stage to the Calculation Stage.
- Open the Exception properties window and check the *Preserve the type and detail of the current exception* box. Notice, that the Exception Type and Detail fields become disabled when the *Preserve* box is checked. This is because the Exception Stage is now set up to only capture and Re-Throw the information from the Exception that is Bubbling up from the Business Object. It is therefore not necessary for the Exception Stage within Recovery Mode, to capture any new information.



In this activity, you will learn basic best practice for the use of Blocks, by experimenting with a Block to cover specific areas of an Action Diagram.

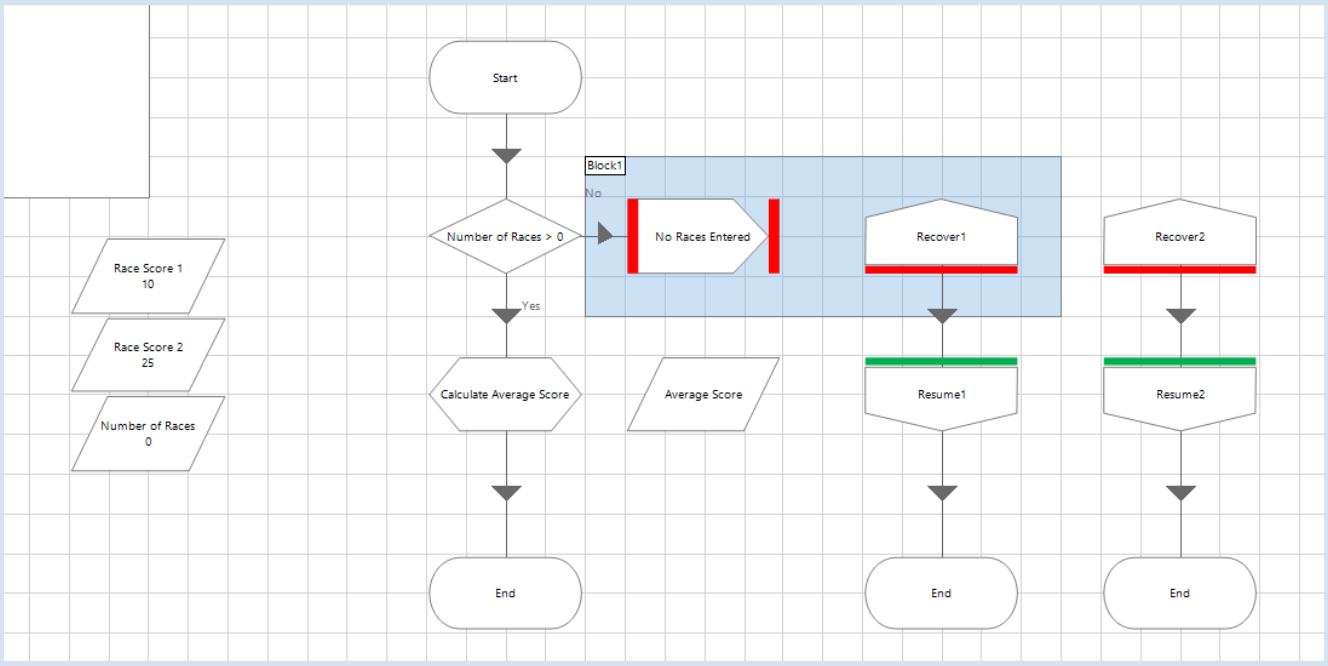
Add a Block and observe its effect.

- For this activity, you'll be working in the *Action 1* Page of the *Racer Score* Business Object.
- Add some new Recovery logic to your Action Diagram (Resume, Recover and End Stages).
- Select the Block Tool in the Stages toolbar.
- *Draw a Block* around the Exception Stage, only.
- Reset and run the Action Page to see the Exception caught by the Recover Stage as normal. When set up in this way, the Block does not provide any benefit to the configuration. As a Block will only work, if it contains a Recover Stage.



Create a second Recovery Mode to control the Exceptions that a Block handles.

- *Copy* and *paste* the Recovery logic on the *Action 1* Page, to create a second Recovery Mode.
- *Stretch* the Block to include the first Recover Stage.
- The first Recover Stage will now only handle Exceptions generated by the Stage sitting inside the same Block.
- Reset and run the Action to see that the Exception is handled by the first Recover Stage. If an Exception were to arise from any of the Stages outside of the Block, then these would now be caught by the second Recover Stage.



Edit the Block to control the functionality of the Recover Stage.

- *Resize* the Block so that it only covers the first Recover Stage.
- Reset and run the Action to see that the first Recover Stage has become redundant because there is no Stage from the main flow inside the Block.